**Docker Enabled Jenkins Image**

* DockerFile for docker enabled Jenkins

FROM jenkins/jenkins:lts

USER root

RUN apt-get update && \

apt-get -y install apt-transport-https \

ca-certificates \

curl \

gnupg2 \

software-properties-common && \

curl -fsSL https://download.docker.com/linux/$(. /etc/os-release; echo "$ID")/gpg > /tmp/dkey; apt-key add /tmp/dkey && \

add-apt-repository \

"deb [arch=amd64] https://download.docker.com/linux/$(. /etc/os-release; echo "$ID") \

$(lsb\_release -cs) \

stable" && \

apt-get update && \

apt-get -y install docker-ce

RUN apt-get install -y docker-ce

RUN usermod -a -G docker jenkins

USER jenkins

* docker build -t jenkinsdocker .

**Steps for Jenkins Pod Creation**

apiVersion: apps/v1

kind: Deployment

metadata:

name: jenkins-deployment

spec:

replicas: 1

selector:

matchLabels:

app: jenkins

template:

metadata:

labels:

app: jenkins

spec:

containers:

- name: jenkins

image: jenkinsdocker

env:

- name: DOCKER\_CONTENT\_TRUST\_SERVER

value: "https://{Notary-Server-IP}:4443"

- name: DOCKER\_CONTENT\_TRUST

value: 1

- name: DOCKER\_CONTENT\_TRUST\_REPOSITORY\_PASSPHRASE

value: "testpass123"

ports:

- containerPort: 8080

volumeMounts:

- name: jenkins-home

mountPath: /home/sandip/jenkins

- name: docker-sock

mountPath: /var/run/docker.sock

volumes:

- name: jenkins-home

emptyDir: {}

- name: docker-sock

hostPath:

path: /var/run/docker.sock

type: File

N.B: Here we are binding host docker socket with the container so that it can run any docker command inside Jenkins pod

**Steps for Implementing Docker Content Trust**

* sudo curl -L "https://github.com/docker/compose/releases/download/1.25.4/docker-compose-$(uname -s)-$(uname -m)" -o /usr/bin/docker-compose
* sudo chmod +x /usr/bin/docker-compose
* yum install git
* git clone https://github.com/theupdateframework/notary.git
* cd notary/
* docker-compose build
* docker-compose up -d
* docker trust key load fixtures/notary-server.key --name notary
* docker trust signer add --key notary-server.crt notary {registry-host}:{registry-port}/admin/demo
* export DOCKER\_CONTENT\_TRUST\_SERVER=https://{Notary-Server-IP}:4443
* export DOCKER\_CONTENT\_TRUST=1
* export DOCKER\_CONTENT\_TRUST\_REPOSITORY\_PASSPHRASE="testpass123"
* docker trust sign {registry-host}:{registry-port}/admin/demo

**Implementing in Jenkins Pipeline**

node('agent') {

print "\*\*\*\* START: SET VARIABLES : START \*\*\*\*"

def mavenHome = tool 'MVN361'

def deployEnv = "${ENVIRONMENT}"

def branchName = "${BRANCH\_NAME}"

def buildNum = "${BUILD\_NUMBER}"

def serviceName = "registration-service"

def imageName = "registration-server"

def imageUrl = "${imageName}:${deployEnv}-${branchName}-${buildNum}"

print "\*\*\*\* END: SET VARIABLES : END \*\*\*\*"

if ( deployEnv == 'prod' ) {

kubeMaster = "{ master-IP }"

dockerRepo = "{ repo-IP }"

} else if ( deployEnv == 'uat' ) {

kubeMaster = "{master-IP}"

dockerRepo = "{repo-IP}"

}

print "\*\*\*\* START: SCM CHECKOUT : START \*\*\*\*"

stage 'CHECKOUT'

checkout scm

print "\*\*\*\* END: SCM CHECKOUT : END \*\*\*\*"

stage 'BUILD APPLICATION'

sh "${mavenHome} -P${deployEnv} clean"

sh "${mavenHome} -P${deployEnv} package"

stage 'BUILD DOCKER IMAGE'

sh "sudo docker build -t ${imageUrl} ."

stage 'TAG/PUSH IMAGE'

sh "sudo docker trust signer add --key notary-server.crt notary ${imageUrl} ${dockerRepo}/${imageUrl}"

sh "sudo docker trust sign ${imageUrl} ${dockerRepo}/${imageUrl}"

sh "sudo docker tag ${imageUrl} ${dockerRepo}/${imageUrl}"

sh "sudo docker push ${dockerRepo}/${imageUrl}"

stage 'DEPLOY TO Kubernetes

sh "chmod +x deploy.sh && ./deploy.sh ${kubeMaster} ${dockerRepo}/${imageUrl} ${serviceName}"

}

N.B: Here we are adding a signer using a certificate and signing a image using it